**Amendments to the Claims:**

The following listing of claims will replace all prior versions, and listings, of claims in the application.

**Listing of Claims:**

Claim 1 (currently amended): A method of maintaining session data in a server system serving a network, said server system including at least one network server and at least one database server, said method comprising the steps of:

(1) storing in a memory that is local to said ~~first~~ at least one network server, session data for a plurality of sessions serviced by said at least one network server;

(2) storing in a database that is local to said at least one database server, copies of said session data for a plurality of sessions serviced by said at least one network server;

(3) performing contemporaneous time out testing of particular session data stored in said memory local to said ~~first~~ at least one network server for one of said plurality of sessions every time a request is received for said particular session data prior to utilizing said particular session data, and not invalidating a copy of said particular session data in said database even if said contemporaneous testing has indicated that the corresponding session has timed out; and

(4) performing an invalidation procedure on said copies of said session data in said database when a load on said at least one database server is below a threshold and at a particular time that is independent of when said contemporaneous testing is performed, wherein said invalidation procedure comprises setting a one-bit flag to indicate that said session data is invalid.

Claim 2 (original): The method of claim 1 wherein said session data comprises an HttpSession object of a Java servlet application program interface (API).

Claim 3 (original): The method of claim 2 wherein said Java servlet APIs are J2EE servlet APIs.

Claim 4 (previously presented): The method of claim 2 wherein step (1) comprises the step of:

(1.1) creating said HttpSession object for a session upon initiation of said session.

Claim 5 (original): The method of claim 4 wherein step (1) further comprises the step of:

(1.2) updating said HttpSession object for said sessions as said session progresses.

Claim 6 (original): The method of claim 5 wherein said server system comprises a plurality of Java Virtual Machines (JVMs) of which different ones of said JVMs may service different http requests corresponding to a single http session and wherein said database is accessible to each of said JVMs.

Claim 7 (currently amended): The method of claim 6 wherein step (1) further comprises the step of:

(1.3) storing said HttpSession object for each session handled by a JVM in a memory local to a network server running said JVM;

and step (2) further comprises the step of:

(2.1) writing a copy of said HttpSession object for each session stored in said ~~said~~ database local to said at least one database server.

Claim 8 (original): The method of claim 7 wherein said plurality of JVMs run on a plurality of network servers.

Claim 9 (original): The method of claim 8 wherein said server system services the World Wide Web.

Claim 10 (original): The method of claim 1 wherein said particular time is a function of a periodic interval.

Claim 11 (original): The method of claim 10 wherein said periodic interval is a day and said particular time is a time of day.

Claim 12 (original): The method of claim 11 wherein said time of day is a time of day that a load on said database is expected to be low.

Claim 13 (previously presented): The method of claim 1 further comprising the steps of:

(5) periodically determining a load on said database; and

wherein said particular time is a function of said determined load and a predetermined interval.

Claim 14 (previously presented) The method of claim 1 wherein said invalidation procedure comprises invalidating all of said copies of said session data stored in said database.

Claim 15 (currently amended) The method of claim 1 wherein said invalidation procedure comprises the steps of:

(4.1) for each copy of said session data in said database, determining if said ~~said~~ session has timed out;

~~(3.2)~~ (4.2) for each session that has timed out, invalidating the corresponding copy of session data in said database.

Claim 16 (currently amended): A server system serving a network comprising:

at least one network server having a local memory;

at least one database server having a database;

a first computer program adapted to store in said memory local to said at least one network server session data for a plurality of sessions serviced by said at least one network server;

a second computer program adapted to store in said database copies of said session data;

a third computer program adapted to perform contemporaneous time out testing of particular session data for one of said plurality of sessions every time a request is received for said particular session data prior to utilizing said particular session data, and further adapted to not invalidating a copy of said particular session data in said database even if said contemporaneous testing has indicated that the corresponding session has timed out; and

a ~~forth~~ fourth computer program adapted to perform an invalidation procedure on said copies of said session data in said database when a load on said at least one database server is below a threshold and at a particular time that is independent of when said contemporaneous testing is performed, wherein said invalidation procedure comprises setting a one-bit flag to indicate that said session data is invalid.

Claim 17 (original): The system of claim 16 wherein said session data comprises an HttpSession object of a Java servlet application program interface (API).

Claim 18 (original): The system of claim 17 wherein said Java servlet APIs are J2EE servlet APIs.

Claim 19 (previously presented): The system of claim 17 wherein said first program creates said HttpSession object for a session upon initiation of said session and updates said HttpSession object for said session as said session progress.

Claim 20 (previously presented): The system of claim 19 further comprising a plurality of Java Virtual Machines (JVMs) of which different ones of said JVMs may service different http requests corresponding to a single session and wherein said memory is accessible to each of said JVMs.

Claim 21 (currently amended): The system of claim 20 wherein said first program stores said HttpSession object for each session handled by a JVM in a memory local to said JVM and wherein said second program writes a copy of said HttpSession object for each http session stored in said ~~said~~ database local to said at least one database server.

Claim 22 (currently amended): The system of claim 21 wherein said at least one network server comprises a plurality of network servers and wherein different ones of said JVMs run on different ones of said plurality of network servers.

Claim 23 (original): The system of claim 22 wherein said server system services the World Wide Web.

Claim 24 (original): The system of claim 16 wherein said particular time is a function of a periodic interval.

Claim 25 (original): The system of claim 24 wherein said periodic interval is a day and said particular time is a time of day.

Claim 26 (original): The system of claim 25 wherein said time of day is a time of day that network traffic involving said server system is expected to be low.

Claim 27 (original): The system of claim 16 further comprising:

a computer program for determining a volume of network traffic involving said server system; and

wherein said particular time is a function of said network traffic involving said server system.

Claim 28 (original): The system of claim 27 wherein said particular time is further a function of a predetermined interval.

Claim 29 (currently amended): The system of claim 16 wherein said ~~forth~~ fourth program invalidates all of said copies of said session data stored in said database at said particular time.

Claim 30 (currently amended): The system of claim 16 wherein, for each copy of session data in said database, said ~~forth~~ fourth program determines if said session has timed out and invalidates the copy of session data corresponding to said sessions that have been determined to have timed out.

Claim 31 (currently amended): A method of maintaining HttpSession objects in a server system serving a network, said server system including a plurality of network servers running a plurality of Java Virtual Machines (JVMs), said method comprising the steps of:

(1) storing in a memory local to a network server running a JVMs HttpSession objects for each session serviced by said JVMs;

(2) storing in a database accessible to all of said JVMs copies of said HttpSession objects for each session serviced by said JVMs;

(3) performing contemporaneous time out testing of a particular HttpSession object every time a request is received for said particular HttpSession object prior to utilizing said particular HttpSession object, and not invalidating a copy of said particular HttpSession object in said database even if said contemporaneous testing has indicated that the corresponding session has timed out; and

(4) performing an invalidation procedure on said copies of said HttpSession objects when a load on said at least one database server is below a threshold and at a particular time that is independent of when said contemporaneous testing is performed, wherein said invalidation procedure comprises setting a one-bit flag to indicate that said session data is invalid.

Claim 32 (previously presented) The method of claim 31 wherein said HttpSession objects conform to the J2EE servlet APIs.

Claim 33 (previously presented): The method of claim 32 wherein step (1) comprises the steps of:

(1.1) creating said HttpSession object for a session upon initiation of said session and storing said HttpSession object in a memory local to a particular one of said JVMs upon initiation of said session;

(1.2) updating said HttpSession object for each said http session in said local memory as said session progresses

and wherein step (2) comprises the steps of:

(2.1) writing a copy of said HttpSession object for each session stored in said local memory to said database upon said creation;

(2.2) updating said copy of said corresponding HttpSession object in said database as said session progresses.

Claim 34 (original): The method of claim 32 wherein said particular time is a function of a periodic interval.

Claim 35 (original): The method of claim 34 wherein said periodic interval is a day and said particular time is a time of day when network traffic involving said server system is expected to be low.

Claim 36 (previously presented) The method of claim 31 further comprising the steps of:

(5) determining a volume of network traffic involving said server system; and

wherein said particular time is a function of said network traffic involving said server system.

Claim 37 (previously presented) The method of claim 31 wherein said invalidation procedure comprises invalidating all of said copies of said HttpSession objects stored in said database at said particular time.

Claim 38 (previously presented) The method of claim 31 wherein said invalidation procedure comprises the steps of:

(3.1) for each copy of an HttpSession object in said database, determining if said corresponding session has timed out; and

(3.2) invalidating each copy of an HttpSession object in said database that has timed out.